



Testing model for web applications using Sink Web Pages and Finite State Machines

¹Doru Anastasiu Popescu, ²Dan Andrei Rusu

¹Department of Computer Sciences and Mathematics

University of Pitesti, Faculty of Informatics and Mathematics, Romania

²School of Electronics and Computer Science

University of Southampton, United Kingdom.

Abstract: An important component in developing web applications is testing. In this paper we will present a testing model, which uses two important notions: sink web pages and the synchronous product of two finite state machines. The sink web pages are used for constructing the client-side finite state machine, respectively the finite state machine used for modeling the server part of the web application. The synchronous product of these two finite state machines is used for generating the tests using different methods.

Keywords: Sink Web Pages; Tag HTML; FSM; IUT; Testing; Web Application

I. Introduction

Web applications represent an important part of software systems, which can solve requirements from different fields. This leads to the use of various resources when developing such applications, like: databases, programming languages, communication services, data transfer, data security, etc. From this perspective, the process of developing a web application consists of various stages and test ways.

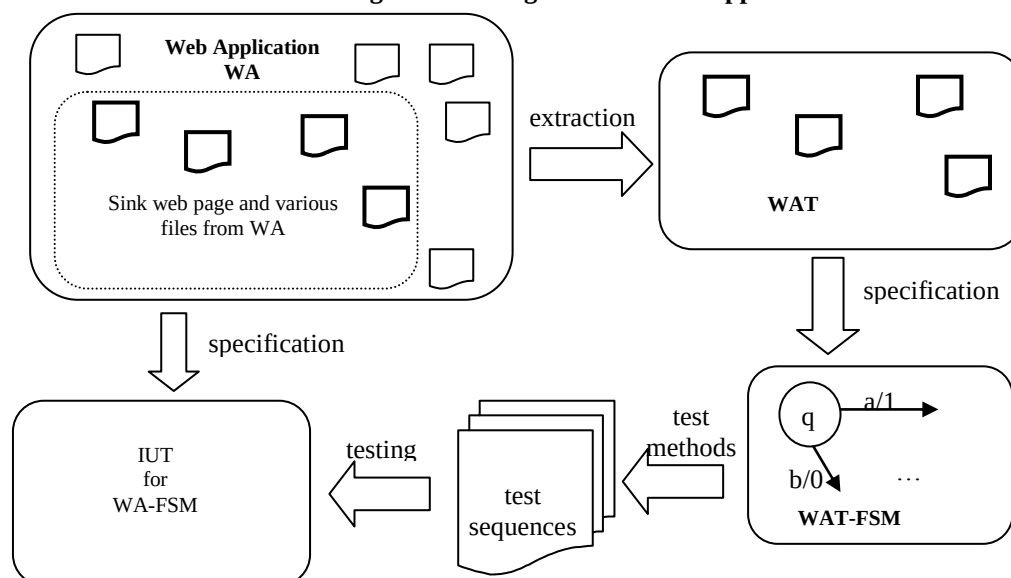
Starting from existing results in testing software systems (as in [1] and [5]), different methods of testing web applications have been developed (as in [2]-[4] and [10]-[12]). In this paper we want to combine two important components of web applications: web applications content (the files that compose the application) and the operating mode (navigability and the way in which components are accessed based on restrictions).

The main idea behind the web applications testing model described in this paper is shown in figure 1 from section II and it consists of selecting some special files from the original application (section III) called sink web pages (notion presented in [7]) and using them for constructing a finite state machine and identifying some test sequences (section IV). These test sequences will be used for testing the specification of the web application IUT (Implementation Under Test) in the process of building and developing.

II. Testing model

The testing model we purpose uses both the source code of the web pages and the web application's operating way. We will use the notions presented in [6]-[9], to determine a web application, noted WAT, with a smaller amount of files than the original web application, noted WA.

Figure 1 Testing model for web application



IUT is Implementation Under Test for Web Application WA.

For the newly obtained web application, WAT, we will construct the specification using finite state machines and we will use them for testing. The test sequences for WAR will be used for testing WA. The testing model is shown in figure 1.

Alongside a great variety of files, web applications may consist of both static and dynamic web pages, this leading to a large number of web pages, depending on the data processed by the server at some point. The specification of a web page can lead to the use of a large amount of states machine. This is why it is useful to use methods of reducing the number of states, methods such as the one presented in this paper.

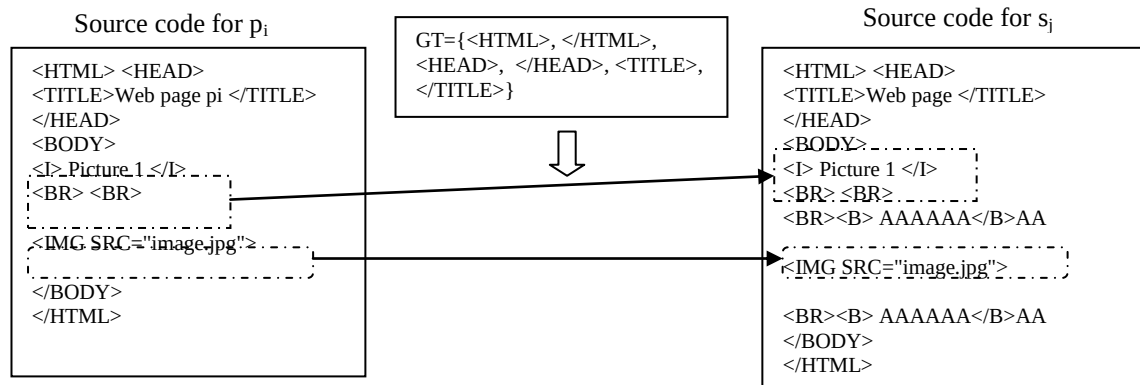
III. Constructing WAT

Next we will consider a web application, noted WA, that has the following set of files: $\{p_1, p_2, \dots, p_n\} \cup \{f_1, f_2, \dots, f_m\}$, $p_1, p_2, \dots, p_n$ being the files that have HTML source code. Next, we note also with $p_1, p_2, \dots, p_n$ the web pages obtained after using those files. Depending of the context, p_i ($1 \leq i \leq n$) will represent either the source code made of tags, either the web page obtained by using a browser.

In order to use as few files as possible in the testing process we identify in the set $\{p_1, p_2, \dots, p_n\}$ some web pages that will form WAT. These files will be called sink web pages and will be determined as shown in [7], [8] using a relationship between web pages. If we note with $s_1, s_2, \dots, s_k$, these web pages, $s_i \in \{p_1, p_2, \dots, p_n\}$, $1 \leq i \leq k$, then WAT will consist of the files: $\{s_1, s_2, \dots, s_k\} \cup \{f_1, f_2, \dots, f_m\}$.

The web pages $s_1, s_2, \dots, s_k$ have the property that for any page p_i from $\{p_1, p_2, \dots, p_n\} \setminus \{s_1, s_2, \dots, s_k\}$, there exists s_j in $\{s_1, s_2, \dots, s_k\}$ with the property that p_i has the tags of its source code included (which cannot be found in a fixed set of tags GT) in s_j . Figure 2 presents an example of such a situation.

Figure 2. Relationship between p_i and s_j

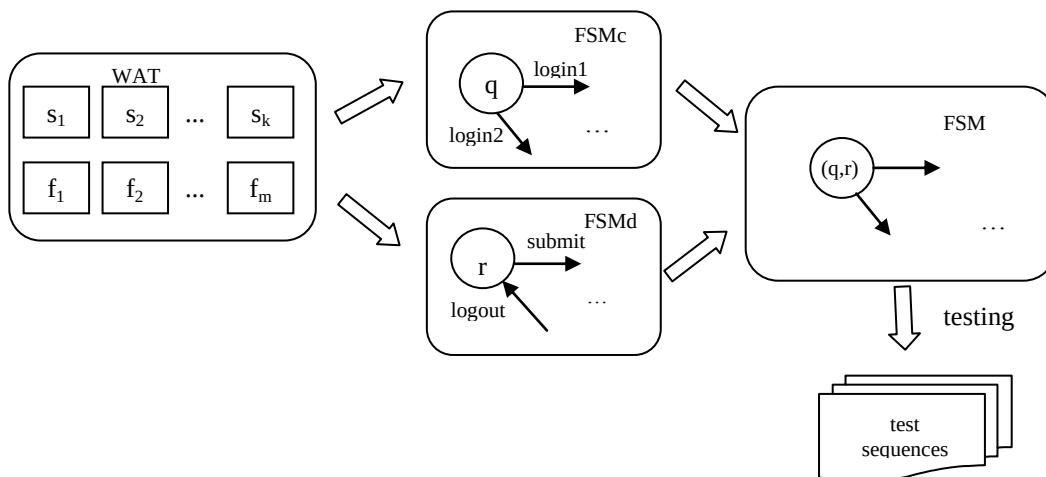


Details, examples and applications of these notions can be found in [6], [7] and [9]. An algorithm for determining sink web pages is presented in [8].

IV. Testing WAT

After constructing the application WAT using files $\{s_1, s_2, \dots, s_k\} \cup \{f_1, f_2, \dots, f_m\}$, its specification using finite state machines follows. This stage consists of: specifying the client side with the finite state machine FSMc, specifying the dynamic part with the finite state machine FSMd, then constructing the synchronous product FSM of the FSMc and FSMd machines. Apart from testing the operating way of WAT, the way to navigate in the application is also tested, details about this second testing being presented in [2], [3] and [10]. The defining way of these finite state machines is shown in [3] and [4]. Figure 3 presents this stage.

Figure 3 WAT Specification using FSM



The finite state machine FSM that specifies WAT application can have a large amount of states, and this is why a successor function is used to determine the pairs of states obtained from the transition operation, as in [4]. The tests obtained using various models (as in [10]-[12]) for FSM will be used to test the implementation of the WA specification. WA specification will be done also using finite state machines, obtaining a finite state machine noted FSM_{WA} . This way, in the process of building and developing the WA application, the states from FSM will be among the states from FSM_{WA} .

V. Testing WA

In figure 1 the testing model for a web application WA was presented, considering its content, the navigation way and the operating way. After constructing the set T with the test sequences for application WAT we will go through the next stages:

1. *Using an existent model (as the ones in [3]) the navigation way in WA is tested.*
2. *The client side of the application WA is modelled, obtaining the finite state machine FSM_1 .*
3. *The server side of the application WA is modelled, obtaining the finite state machine FSM_2 .*
4. *The synchronous product FSM_{WA} of the machines FSM_1 and FSM_2 is constructed.*
5. *Using the test sequences from T the implementation IUT_{WA} of the specification for the application WA is tested.*

The advantages of this method consist of the fact that for generating the test sequences a finite state machine FSM with a smaller size (smaller number of states and transitions) than FSM_{WA} is used. Generating tests for WAT, used also with WA, can be done using a large variety of methods, as the ones presented in [1], [5] and [10]-[12].

VI. Conclusion

With this paper we wished to present a testing model for web applications considering both the constructive elements and the aspects related to navigation and functionality. In the papers used as references are presented theoretical notions, which demonstrate the correctness of every stage of the presented model. The notions used in section III have applicability in other fields as well, such as the rank of a web application, searching in web applications, comparing web applications and determining a way of calculating their similarity (as in [6]-[9]).

Testing software systems is a well developed field with theoretical models that can be adapted depending on the specification of the applications (as in [1] and [5]). Web applications are a particular case of software applications, characterized by the variety of the languages used for their creation, the connection between components, working with databases and the multitude of applications that have to be used at a certain point. All these aspects lead to a rise in the difficulty of testing, the existing models managing only partially to test the application. In sections IV and V deterministic finite state machines have been used for specification and testing (as in [2]-[4] and [10]-[12]).

VII. References

- [1] Aditya P. Mathur, "Foundations of Software Testing", Person Education Publication, 2008.
- [2] Anneliese A. Andrews, Jeff Offutt, Roger T. Alexander, "Testing Web applications by modeling with FSMs", *Softw Syst Model* (2005) 4: 326-345
- [3] Bo Song, Huaikou Miao, Shengbo Chen, "Modeling Web Browser Interactions and Generating Tests", *International Conference on Computational Intelligence and Security*, 2008, IEEE, 399-404.
- [4] Bo Song, Shengwen Gong, Shengbo Chen, "Model Composition and Generating Tests for Web Applications", *International Conference on Computational Intelligence and Security*, 2011, IEEE, 568-572.
- [5] David Lee, Mihalis Yannakakis, *Principles and Methods of Testing Finite State Machines - A Survey*, AT&T Bell Laboratories, 2000.
- [6] Doru Popescu Anastasiu, Catrinel Maria Danauta, "Similarity measurement of web sites using sink web pages", *34th International Conference on Telecommunications and Signal Processing, TSP 2011*, August 18-20, Budapest, Hungary, *IEEE Xplore*, 24-26.
- [7] Doru Anastasiu Popescu, "Sink Web Pages in Web Application", *IAPR TC3 Workshop on Partially Supervised Learning (PSL2011)*, September 15-16, 2011, University of Ulm, Germany, *Proceedings LNCS*, Springer 2012, 154-158.
- [8] Doru Anastasiu Popescu, Danauta Catrinel Maria, "Verification of the web applications using sink web pages", *International Journal of Computer Science Research and Application*, issue 1, 2012, pag. 63-68.
- [9] Doru Anastasiu Popescu, Dragos Nicolae, "Similarity Measurement of Web Site", *SOFA 2012, 5th International Workshop Soft Computing Applications*, *Proceedings LNCS*, Springer 2013, 349-356
- [10] Joumana Dargham, Sukaiana Al-Nasrawi, "FSM Behavioral Modeling Approach for Hypermedia Web Applications: FBM-HWA Approach", *AICT/ICIW 2006*, IEEE
- [11] Kulvinder Singh, Rakesh Kumar, Iqbal Kaur, "Testing Web Based Applications using Finite State Machines Employing Genetic Algorithm", *IJEST*, 2010, 6931-6941
- [12] Kulvinder Singh and Semmi, "Finite State Machine based Testing of Web Applications", *IJSWS*, 2013, 41-46